please enter
N.E.O.
8/4/05

PTO/SB/21 (09-04)
Approved for use through 07/31/2006. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

TRANSMITTAL FORM

(to be used for all correspondence after initial filing)

Total Number of Pages in This Submission	

Application Number	09/755,286
Filing Date	01/03/2001
First Named Inventor	Dirk Coldewey
Art Unit	2177
Examiner Name	Harold E. Dodds, Jr.
Attorney Docket Number	

ENCLOSURES *(Check all that apply)*

- ☐ Fee Transmittal Form
  - ☐ Fee Attached
- ☑ Amendment/Reply
  - ☐ After Final
  - ☐ Affidavits/declaration(s)
- ☐ Extension of Time Request
- ☐ Express Abandonment Request
- ☐ Information Disclosure Statement
- ☐ Certified Copy of Priority Document(s)
- ☐ Reply to Missing Parts/ Incomplete Application
  - ☐ Reply to Missing Parts under 37 CFR 1.52 or 1.53
- ☑ Drawing(s)
- ☐ Licensing-related Papers
- ☐ Petition
- ☐ Petition to Convert to a Provisional Application
- ☐ Power of Attorney, Revocation Change of Correspondence Address
- ☐ Terminal Disclaimer
- ☐ Request for Refund
- ☐ CD, Number of CD(s)_____
  - ☐ Landscape Table on CD
- ☐ After Allowance Communication to TC
- ☐ Appeal Communication to Board of Appeals and Interferences
- ☐ Appeal Communication to TC (Appeal Notice, Brief, Reply Brief)
- ☐ Proprietary Information
- ☐ Status Letter
- ☐ Other Enclosure(s) (please identify below):

Remarks

Please see separate remarks page between markup copy of application and the non-markup copy of the transmission.

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Firm Name			
Signature	/Dirk Coldewey/		
Printed name	Dirk Coldewey		
Date	7/27/2005	Reg. No.	

CERTIFICATE OF TRANSMISSION/MAILING

I hereby certify that this correspondence is being facsimile transmitted to the USPTO or deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on the date shown below:

Signature	/Dirk Coldewey/		
Typed or printed name	Dirk Coldewey	Date	7/27/2005

(Replacement Sheet ~~drawings separated from claims section~~)

```
Traverse( forest_ptr forest )
  {
    /* local variables */
    stack stacks[PipeDepth];  /* PipeDepth stacks */
    tree_ptr n;
    int i, trees_left = PipeDepth;
    struct {
      tree_ptr node;
      stack_ptr stack;
    } traversal[PipeDepth];     /* traversal state descriptor */

    /* prologue */
    for ( i=0; i<PipeDepth; i++ ) {
      traversal[i].node = forest->root[i];
      traversal[i].stack = &stack[i];
      PREFETCH(forest->root[i], sizeof(forest->root[i]));
    }

    /* steady state */
    while ( trees_left ) {
      for ( i=0; i<trees_left; i++ ) {
        if ( traversal[i].node->left ) {
          traversal[i].stack->push( traversal[i].node->left );
          traversal[i].node = traversal[i].node->left;
        } else {
          n = traversal[i].stack->pop();
          if ( n == NULL ) {     /* done with tree i */
            trees_left--;
            if ( i != trees_left )
              SWAP( &traversal[i], &traversal[trees_left] );
          }
          process( n );
          traversal[i].node = n->right;
        }
        PREFETCH( traversal[i].node );
      }
    }
  }
```

Figure 7: Example of a Pipelined Tree Traversal

(Replacement Sheet ~~– drawings separated from claims~~)

```
Traverse( tree_ptr tree )
  {
    /* local variables */
        . . .

    /* level-order traversal prologue */
    PREFETCH( tree->root );
    enqueue( src_queue, tree->root );
    for ( i=0, accumulating=true; accumulating; i++ ) {
      n = dequeue(src_queue);
      if ( n == NULL )
        return;              /* we're done */
      process(n->data);

      if ( n->left != NULL ) {
        PREFETCH( n->left );
        enqueue( dst_queue, n->left );
      }
      if ( n->right != NULL ) {
        PREFETCH( n->right );
        enqueue( dst_queue, n->right );
      }
      if ( src_queue->size + dst_queue->size < PipeDepth ) {
        if ( i >= src_queue->size )
          SWAP( src_queue, dst_queue );
      } else {
        accumulating = false;
        while ( src_queue->size > 0 ) {
          traversal[trees_left].node = dequeue( src_queue );
          traversal[trees_left].stack = stack[trees_left];
          trees_left++;
        }
        while ( dst_queue->size > 0 ) {
          traversal[trees_left].node = dequeue( dst_queue );
          traversal[trees_left].stack = stack[trees_left];
          trees_left++;
        }
      }
    }

    /* steady state loop */
        . . .
```

Figure 8: Example of a pipelined level-order tree traversal.